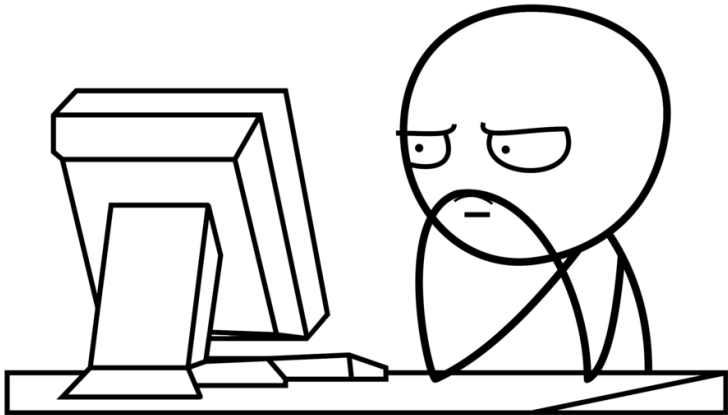


Universität Stuttgart
Institut für Luftfahrtssysteme



Tutorium 2:

Einfache IO

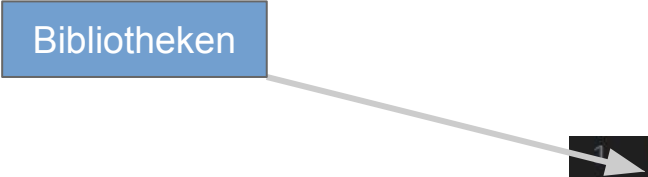
(Ein- und Ausgabe)

Übersicht für heute:

- Präsentation zur Erklärung von Bibliotheken und Ein- und Ausgabe über die Konsole
- Aufgabe zum selbst Coden
- Lösung mit Erklärung

Bibliotheksfunktionen

Bibliotheken



```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main (void)
5 {
6     printf ("Hello World!\n");
7     return EXIT_SUCCESS;
8 }
```

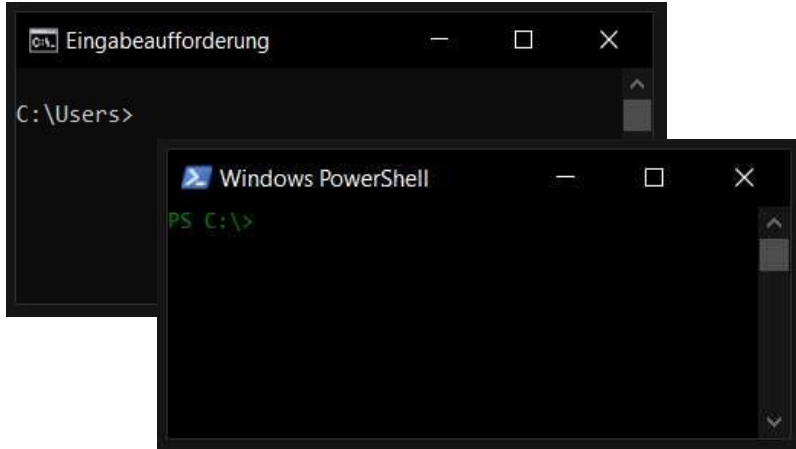
Bibliotheksfunktionen

Was sind Bibliotheken?

- Sammlung von häufig verwendeten Funktionen
- Werden an den Programmcode angehängt
- Funktionen müssen nicht jedes Mal neu geschrieben werden

- Häufig verwendete Bibliotheken:
 - › Ein- & Ausgabe <stdio.h>
 - › Speicherverwaltung & Prozesssteuerung <stdlib.h>
 - › Mathe <math.h>
 - › Zeichenketten <string.h>

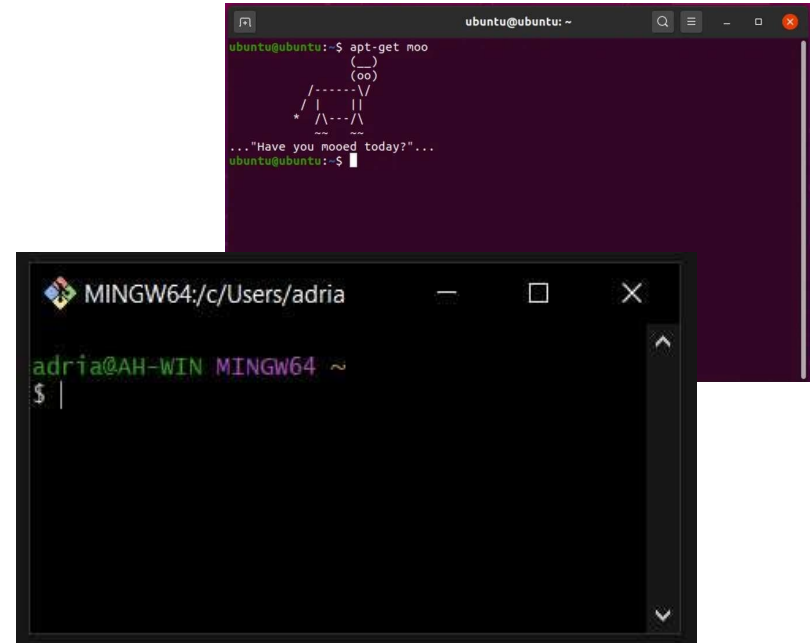
Ein- und Ausgabe über die Konsole



The image shows two overlapping Windows console windows. The background window is titled 'Eingabeaufforderung' (Command Prompt) and shows the prompt 'C:\Users>'. The foreground window is titled 'Windows PowerShell' and shows the prompt 'PS C:\>'. Both windows have a dark theme.

```
Eingabeaufforderung
C:\Users>

Windows PowerShell
PS C:\>
```

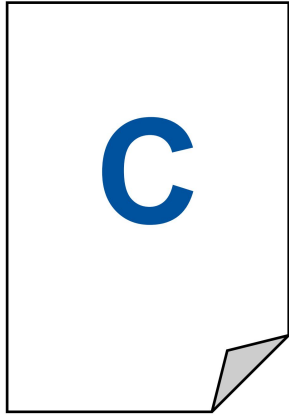


The image shows two overlapping terminal windows. The background window is an Ubuntu terminal with a purple background, showing the command 'apt-get moo' and its output, which includes a ASCII art cow and the text '...Have you mooed today?'. The foreground window is a MINGW64 console window with a dark theme, showing the prompt 'adria@AH-WIN MINGW64 ~' and a dollar sign '\$' followed by a cursor.

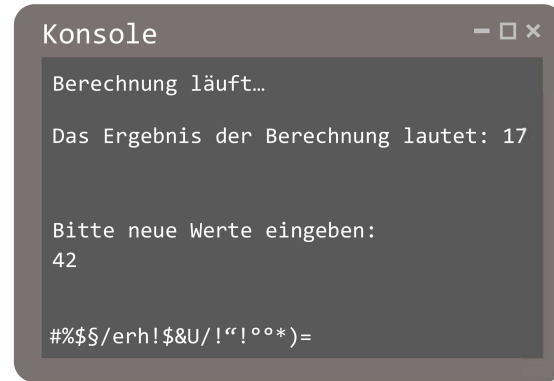
```
ubuntu@ubuntu:~$ apt-get moo
(oo)
 /---\
*  /---\
  /---\
 /---\
/---\
...Have you mooed today?...
ubuntu@ubuntu:~$

MINGW64:/c/Users/adria
adria@AH-WIN MINGW64 ~
$ |
```

Ein- und Ausgabe über die Konsole



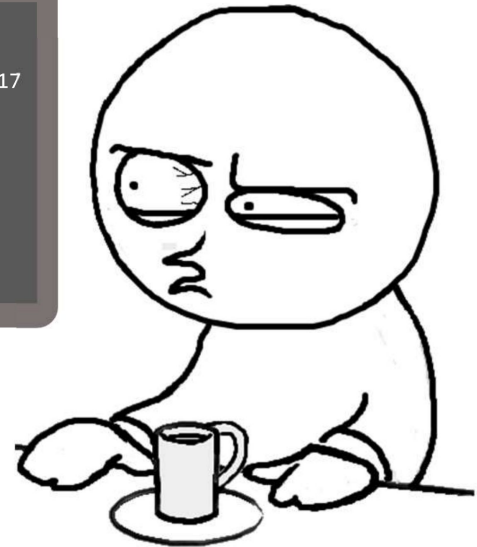
Ausgabe



```
Konsole
Berechnung läuft...
Das Ergebnis der Berechnung lautet: 17

Bitte neue Werte eingeben:
42

#%$$/erh!$&U/!“!°°*)=
```



Ausgabe mit printf

Semikolon hinter jedem Befehl!

Syntax:

```
printf(FORMAT, PARAMETER);
```



Also...

```
Gib aus(so formatiert, das hier);
```

Beispiel:

```
printf("%d", 10);
```

Also...

```
Gib aus(eine Ganzzahl, Zahl 10);
```

Ausgabe mit printf: Parameter und Datentypen

- Die Parameter ermöglichen die Ausgabe von Variablenwerten
- Beispiel im Code:

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    int x = 3;
    printf("Die Zahl ist: %d", x);
    return EXIT_SUCCESS;
}
```


Ausgabe mit printf

printf ermöglicht:

- Formatierte Ausgabe
 - › Gerundet
 - › Mit definierten Abständen
 - › Verschiedene Zahlensysteme
 - › Exponentialschreibweise
- Ausgabe mehrerer Datentypen gleichzeitig

Hinweis: Formatieroptionen werden im Merkblatt erklärt!

```
Hello World!
```

```
10
```

```
255
```

```
000100020003
```

```
1    2    3
```

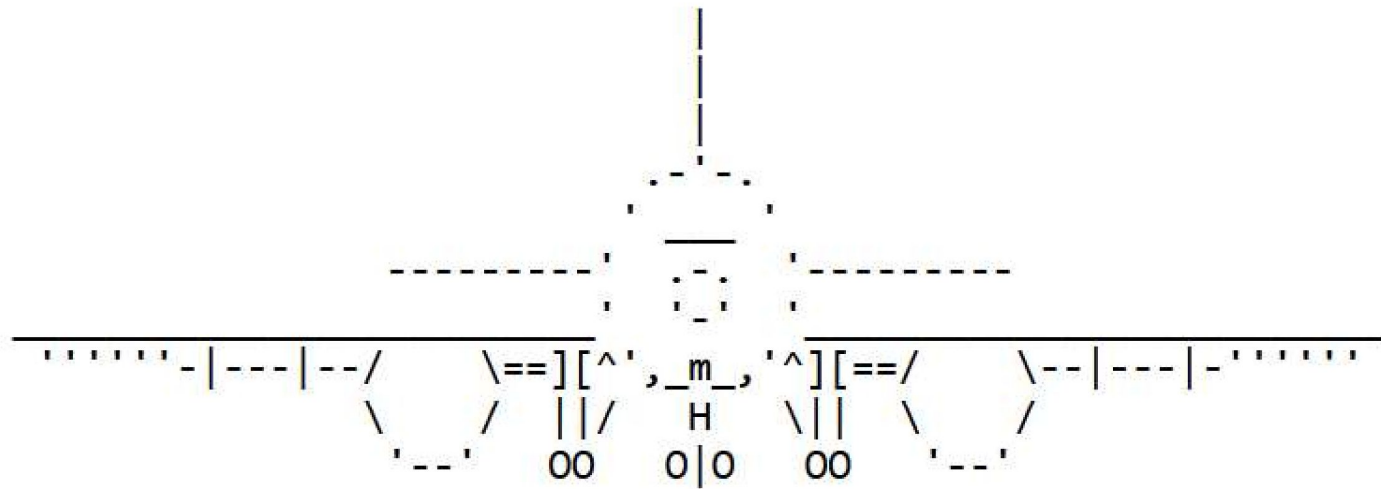
```
3.1415927
```

```
3.14
```

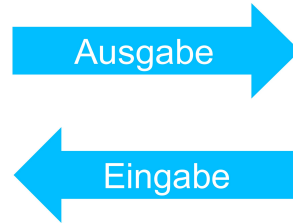
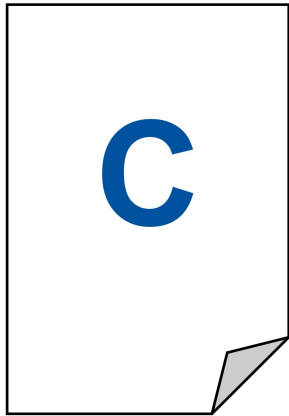
```
c
```

```
Airbus A380
```

Ausgabe mit printf



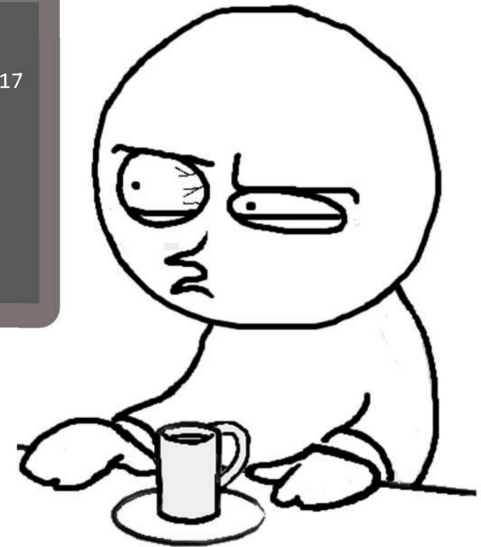
Ein- und Ausgabe über die Konsole



```
Konsole
Berechnung läuft...
Das Ergebnis der Berechnung lautet: 17

Bitte neue Werte eingeben:
42

#%$$/erh!$&U/!“!°°*)=
```



Eingabe mit scanf

Syntax:

```
scanf(FORMAT, SPEICHERORT);
```

Also...

```
Lese ein(so formatiert, speichere hier);
```



Das „&“ ist wichtig!
Nicht vergessen*

Beispiel:

```
scanf("%d", &x);
```

Also...

```
Lese ein(eine Ganzzahl, in Variable x);
```

*außer bei Strings (Text), warum
lernt ihr noch

Beispiel im Code

Code von vorhin:

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
int x = 3;
    printf("Die Zahl ist: %d", x);
    return EXIT_SUCCESS;
}
```

Beispiel im Code

Jetzt:

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    int x;
    printf("Bitte eine Zahl eingeben: ");
    scanf("%d", &x);
    printf("Die Zahl ist: %d", x);
    return EXIT_SUCCESS;
}
```

Einfache Ein- und Ausgabe

Aufgabe

- Aufgabenstellung und Erklärungsblatt sowie Codegerüst auf ILIAS
- Erklärungsblatt durchlesen
- Codegerüst bearbeiten

Am Ende kurze Besprechung, Musterlösung wird nach dem letzten Tutorium veröffentlicht.

Vielen Dank für eure Aufmerksamkeit

- Pushen nicht vergessen!
 - › `git add .`
 - › `git commit -m "Kommentar"`
 - › `git push`

Wichtig: Ihr müsst euch im richtigen Ordner (Software-Ordner) befinden, um pushen zu können! Ihr könnt euch mit `cd` (change directory) bewegen.

Allgemein: Semikolon und Speichern nicht vergessen!

Viel Spaß :D